

IN THE CLAIMS:

Please cancel claims 24 and 25 and add new claims 29-36. Please amend the remaining claims as follows:

1. (Previously amended) A cache, comprising a plurality of independently addressable cachelets, each cachelet to provide data responsive to independent load requests in a single clock cycle.
2. (Original) The cache of claim 1, wherein the cache is a member of a multiple layer cache system.
3. (Original) The cache of claim 1, further comprising an address manager, coupled to an input of the cache and to each of the cachelets.
4. (Original) The cache of claim 1, wherein each cachelet comprises:  
a plurality of cache entries, each cache entry having tag and data fields,  
an address decoder coupled to an address input and to the cache entries, and  
a tag comparator coupled to the address input and to the tag fields of the cache entries.
5. (Original) The cache of claim 4, wherein the address inputs of each of the cachelets are independent from each other.
6. (Original) A processing system, comprising:  
the cache of claim 1,  
an instruction decoder,  
an address manager coupled to the instruction decoder, and  
a plurality of load units coupled to the address manager, each of the load units coupled to a respective one of the cachelets.
7. (Original) A processing system, comprising:  
the cache of claim 1,  
an instruction decoder,  
a plurality of load units coupled to the instruction decoder,

an interconnect providing dynamic communication between the load units and the cachelets.

8. Cancelled

9. (Previously amended) The cache assignment method of claim 10, the method further comprising forwarding any data requests associated with non-conflicting cachelet pointers to cachelets identified by the respective pointers.

10. (Previously amended) A cache assignment method, comprising:  
receiving plural data requests, each associated with respective cachelet pointers,  
determining whether any of the cachelet pointers conflict with any other cachelet  
pointers,

if a conflict occurs among cachelet pointers, forwarding one of the data requests  
associated with a conflicting cachelet pointer to the identified cachelet, and

reassigning data requests associated with remaining conflicting cachelet pointers to  
unused cachelets.

11. (Original) The cache assignment method of claim 10, wherein multiple data requests  
having a common set address are forwarded to different cachelets.

12. (Previously amended) The cache assignment method of claim 10, further comprising:  
determining whether any of the cachelet pointers are valid,  
forwarding data requests having valid, non-conflicting cachelet pointers to the addressed  
cachelet, and  
assigning data requests of non-conflicting cachelet pointers to unused cachelets  
according to a default assignment scheme.

13. (Previously amended) The cache assignment method of claim 10, wherein copies of a  
single data item may be stored in multiple cachelets.

14. (Original) A cache assignment method, comprising:  
receiving plural data requests and associated cachelet pointers, the cachelet pointers  
addressing one of a plurality of cachelets within a cache,

determining whether any of the cachelet pointers conflict with any other cachelet pointers,

forwarding non-conflicting data requests to a cachelet identified by the cachelet pointer,

for the conflicting data requests, forwarding one of the conflicting data requested to the identified cachelet and

reassigning remaining conflicting data requests to unused cachelets.

15. (Previously amended) The cache assignment method of claim 14 wherein the data requests are associated with respective cachelet pointers, the method further comprising:

determining whether any of the cachelet pointers are valid, and

assigning remaining data requests to unused cachelets according to a default assignment scheme.

16. (Original) A cache assignment method, comprising:

receiving plural data requests and associated cachelet pointers, the cachelet pointers addressing one of a plurality of cachelets within a cache,

determining whether any of the cachelet pointers are valid,

forwarding data requests having valid cachelet pointers to the addressed cachelet, and

assigning remaining data requests to unused cachelets according to a default assignment scheme.

17. (Original) The cache assignment method of claim 16, further comprising:

determining whether any of the cachelet pointers conflict with any other cachelet pointers,

forwarding any data requests associated with non-conflicting cachelet pointers to cachelets identified by the respective pointers.

18. (Original) The cache assignment method of claim 16, further comprising:

if a conflict occurs among cachelet pointers, forwarding one of the data requests associated with a conflicting cachelet pointer to the identified cachelet, and

reassigning data requests associated with remaining conflicting cachelet pointers to unused cachelets.

19. (Previously amended) A cache comprising:

a plurality of independently addressable cachelets,  
means for distributing independent loads to each of the cachelets in a single clock cycle.

20. (Original) A cache system, comprising:  
the cache of claim 19 provided as a first layer of cache, and  
a second layer of cache to receive a load that misses the cachelet to which it was assigned.

21. (Original) The cache system of claim 20, wherein the second layer of cache is a system memory.

22. (Previously Added) A cache system comprising:  
a first layer of cache, comprising a plurality of independently addressable cachelets and means for distributing multiple loads among the cachelets in a single clock cycle, and  
a second layer of cache to receive a load that misses the cachelet to which it was assigned.

23. (Previously Added) The cache system of claim 22, wherein the second layer of cache is a system memory.

24-25. Cancelled.

26. (Previously Added) A cache assignment method, comprising:  
receiving plural data requests, wherein the data requests are associated with respective cachelet pointers,  
determining whether any of the cachelet pointers are valid,  
forwarding data requests having valid cachelet pointers to the addressed cachelet, and  
assigning remaining data requests to unused cachelets according to a default assignment scheme.

27. (Previously Added) The cache assignment method of claim 26, wherein each data request includes a cachelet pointer, the method further comprising forwarding any data requests associated with non-conflicting cachelet pointers to cachelets identified by the respective pointers.

28. (Previously Added) The cache assignment method of claim 26, the method further comprising:

receiving plural data requests, each associated with respective cachelet pointers,

determining whether any of the cachelet pointers conflict with any other cachelet pointers,

if a conflict occurs among cachelet pointers, forwarding one of the data requests associated with a conflicting cachelet pointer to the identified cachelet, and

reassigning data requests associated with remaining conflicting cachelet pointers to unused cachelets.

Please add the following new claims:

29. (New) The cache of claim 1, wherein the cachelets each receive independent addresses in a common clock cycle.

30. (New) The cache assignment method of claim 10, further comprising forwarding the reassigned data requests in parallel with the other forwarded data requests.

31. (New) The cache assignment method of claim 14, further comprising forwarding the reassigned data requests in parallel with the non-conflicting data requests and the one conflicting data request.

32. (New) The cache assignment method of claim 14, further comprising forwarding the assigned data requests to the unused cachelets in parallel with the other forwarded data requests.

33. (New) The cache assignment method of claim 26 further comprising forwarding the assigned data requests in parallel with the forwarding of data requests having valid cachelet pointers.

34. (New) A cache management method, comprising:

receiving plural data requests, and

in unison:

directing one data request to each cachelet within the cache,

for each cachelet, determining if the respective data request hits the cachelet and, if so, performing the respective data request.

*cl  
apl*  
35. (New) The cache management method of claim 34, further comprising posting a write request simultaneously to all of the cachelets.

36. (New) The cache management method of claim 34, further comprising, if a respective data request does not hit its cachelet, directing the data request to another cache layer.

---